CHALMERS
UNIVERSITY OF TECHNOLOGY

Surrogate-based optimisation using adaptively scaled radial basis functions

Citation for the original published paper (version of record):
Urquhart, M., Ljungskog, E., Sebben, S. (2020). Surrogate-based optimisation using adaptively scaled radial basis functions. Applied Soft Computing Journal, 88.
http://dx.doi.org/10.1016/j.asoc.2019.106050

Applied Soft Computing Journal 88 (2020) 106050

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Surrogate-based optimisation using adaptively scaled radial basis functions

Magnus Urquhart *, Emil Ljungskog, Simone Sebben

Chalmers University of Technology, Gothenburg, Sweden

ARTICLE INFO

Article history:
Received 27 February 2019
Received in revised form 21 November 2019
Accepted 25 December 2019
Available online 2 January 2020

Keywords:
Evolutionary algorithm
Black box optimisation
Bayesian optimisation
Benchmarking
Optimisation
Global optimisation
Gradient-free
Aerodynamics
Surrogate model
Proper Orthogonal Decomposition
Radial Basis Function interpolation
Latin Hypercube Sampling

ABSTRACT

Aerodynamic shape optimisation is widely used in several applications, such as road vehicles, aircraft and trains. This paper investigates the performance of two surrogate-based optimisation methods; a Proper Orthogonal Decomposition-based method and a force-based surrogate model. The generic passenger vehicle DrivAer is used as a test case where the predictive capability of the surrogate in terms of aerodynamic drag is presented. The Proper Orthogonal Decomposition-based method uses simulation results from topologically different meshes by interpolating all solutions to a common mesh for which the decomposition is calculated. Both the Proper Orthogonal Decomposition- and force-based approaches make use of Radial Basis Function interpolation. The Radial Basis Function hyperparameters are optimised using differential evolution. Additionally, the axis scaling is treated as a hyperparameter, which reduces the interpolation error by more than 50% for the investigated test case. It is shown that the force-based approach performs better than the Proper Orthogonal Decomposition method, especially at low sample counts, both with and without adaptive scaling. The sample points, from which the surrogate model is built, are determined using an optimised Latin Hypercube sampling plan. The Latin Hypercube sampling plan is extended to include both continuous and categorical values, which further improve the surrogate's predictive capability when categorical design parameters, such as on/off parameters, are included in the design space. The performance of the force-based surrogate model is compared with four other gradient-free optimisation techniques: Random Sample, Differential Evolution, Nelder–Mead and Bayesian Optimisation. The surrogate model performed as good as, or better than these algorithms, for 17 out of the 18 investigated benchmark problems.

1. Introduction

Optimisation is an ongoing field of research and aims to find the best solution to a specific problem while obeying some constraints. Given that resources and time are a limitation, it is often not feasible to try all possible design combinations. As a consequence, it is practically not possible to determine if a given solution is the global optimum.

The field of optimisation is often split into two categories, gradient-based and gradient-free techniques. Gradient-based optimisation relies on the gradient information to determine in which direction to search for a better design. Gradient-based methods perform well for unimodal problems; however, they risk converging to a local optimum if the problem exhibits several minima [1]. Gradient-based methods require gradient information to be available, something which is not possible when dealing with categorical, or discrete, values. When working with gradient-based optimisation of numerical aerodynamic simulations, it is essential to use robust mesh-deforming strategies so that each simulation produces meaningful results. Without such strategies, the optimisation method may exploit mesh discretisation errors leading to significant discrepancies between simulated and actual performance [2].

Gradient-free optimisation can be adopted with ease into existing practices; however, a major limitation is the often high number of objective function evaluations needed. The cost of one objective function evaluation, for example, a Computational Fluid Dynamics (CFD) simulation, can be expensive. Joly et al. [3] used low-cost Reynolds-averaged Navier–Stokes (RANS) CFD simulations to reduce the computational effort when optimising the vane–rotor shock interaction with the Evolutionary Algorithm (EA) Differential Evolution (DE). Massaro and Benini [4] used an Artificial Neural Network as a global surrogate in a surrogate-assisted evolutionary framework for optimisation to improve the convergence speed of the optimisation problem.

In this work, the expensive cost of running a multitude of CFD simulations is mitigated by the use of a surrogate model

* Corresponding author.
E-mail addresses: magnus.urquhart@chalmers.se (M. Urquhart), emil.ljungskog@chalmers.se (E. Ljungskog), simone.sebben@chalmers.se (S. Sebben).

https://doi.org/10.1016/j.asoc.2019.106050

Nomenclature

Abbreviations

BO	Bayesian Optimisation
CFD	Computational Fluid Dynamics
DE	Differential Evolution
DOE	Design Of Experiments
EA	Evolutionary Algorithm
LHC	Latin Hypercube
MAE	Mean Absolute Error
NM	Nelder–Mead
PCA	Principle Component Analysis
POD	Proper Orthogonal Decomposition
RANS	Reynolds Averaged Navier Stokes
RBF	Radial Basis Function
RS	Random Sample
SM	Surrogate Model

Symbols

λ	Ridge regression factor
$\mathbf{X}$	Space and design-dependent flow field matrix
$\phi_i(\mathbf{r})$	ith POD mode
$\tilde{f}_i$	ith surrogate model prediction
ε	RBF width
ξ_i	ith RBF weight
$a_i(d)$	ith design-dependent mode coefficient
C_DA	Drag area
C_LA	Lift area
e_i	Interpolation error at the ith sample
$E_{RMS,LOO}$	Root Mean Squared Leave One Out Error
L^2	Euclidean norm
w_i	ith Radial Basis Function
$x_{i,scaled}$	Scaled design variable
c_j	Dimension scaling factor

(SM) when searching for new designs. The surrogate model is comparatively cheap to evaluate and create; however, the new designs found by exploring the surrogate model is subject to the accuracy of the model. Strategies need to be employed that increase the accuracy of the surrogate while exploring promising regions of the design space. These strategies typically involve balancing exploration and exploitation. Exploration involves sampling locations that improve the surrogate model's accuracy by exploring the design space, for example, by sampling sparse areas, while pure exploitation is achieved by sampling the best-predicted design location. When new designs are validated, they are added to the initial set and the surrogate model is built again. One such strategy was developed by Goel et al. [5], who used an ensemble of surrogates to determine regions where the standard deviation between the surrogates was large to find regions of high uncertainty to explore the design space. However, they noted that regions of low standard deviation do not necessarily correlate with low uncertainty. Another approach developed by Sun et al. [6] uses a combination of two metaheuristic optimisation algorithms, one focusing on exploration and the other on local refinement of the solution. The combined approach yielded high performance for problems of large dimensionality, in this case, up to 200 dimensions.

Two surrogate models are compared in this work, a Proper Orthogonal Decomposition (POD)-based surrogate and a force-based surrogate. Several authors have investigated POD-based optimisation [7–11]. As an example, Miretti et al. [9] found that the use of a POD-based surrogate model reduced the number of computations needed by 30% to 40% compared with other gradient-free methods when optimising the drag of a road vehicle. In the POD-based approach, the entire flow field is used for each input sample and an entire flow field is given as the output. The input to the force-based surrogate model is a scalar and the output is an estimate of the same scalar in a new design point. Note that the scalar of interest can be any quantity, not only a force.

The focus of this paper is on the accuracy and performance of the surrogate model. The accuracy of the method is evaluated using CFD simulations of the generic vehicle DrivAer, developed by the Technical University of Munich [12]. The surrogate accuracy is investigated using three design variables; roof angle, diffuser height and front-wheel deflector on/off. The surrogate model's performance, when used in optimisation, is benchmarked with nine test functions with and without added noise.

The main contributions of this work are:

(1) A new axis scaling parameter has been implemented in the Radial Basis Function (RBF) based surrogate model which improved the surrogate model fit. The improvement is pronounced when there is a significant difference in sensitivity between the different input dimensions in regards to the function output.
(2) A new method of creating optimised Latin Hypercube sampling plans, including categorical design spaces, was developed which halved the interpolation error when combined with adaptively scaling RBFs.
(3) It was shown that the inclusions of a POD-based method did not improve the interpolation accuracy over a force-based surrogate model for the investigated aerodynamic test cases.

2. Methodology

2.1. Proper orthogonal decomposition

Proper Orthogonal Decomposition is a modal decomposition technique that extracts orthogonal modes from the field of interest by optimising the mean square field variable captured by each mode. If the field variable of interest is velocity, this is analogous to capturing the most amount of kinetic energy using the least amount of modes. POD is also known as the Karhunen–Loève procedure or Principle Component Analysis (PCA) and has been used in many applications such as statistics, turbulent flows, reduced-order modelling and optimisation. POD was first introduced as a data processing technique in fluid flows by Lumley [13]. A brief introduction to the POD technique is given below. More information can be found in the works by Taira et al. [14] and Muld et al. [15].

It is assumed that the flow can be decomposed as

$$\mathbf{X}(\mathbf{r}, d) = \sum_{i=1}^{m} a_i(d)\phi_i(\mathbf{r}) \tag{1}$$

where $\phi_i(\mathbf{r})$ is the ith POD mode and $a_i(d)$ are the design-dependent mode coefficients containing the design information. $\mathbf{X}$ are the space and design-dependent flow variables arranged as

$$\mathbf{X} = \begin{bmatrix} x_{1,d_1} & x_{1,d_2} & \dots & x_{1,d_m} \\ x_{2,d_1} & x_{2,d_2} & \dots & x_{2,d_m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,d_1} & x_{n,d_2} & \dots & x_{n,d_m} \end{bmatrix}$$

with n data points and m snapshots, where one snapshot corresponds to one CFD simulation. The variable matrix $\mathbf{X}$ is built from the pressure and velocity fields in the entire computational domain. As mentioned, the POD method seeks to find the orthogonal basis vectors that optimally represent the data in a mean squared, or L^2 sense. It is possible to find these basis vectors by solving the eigenvalue problem

$$\mathbf{Y}\psi_i = \lambda_i \psi_i, \quad \mathbf{Y} = \mathbf{X}\mathbf{X}^T \tag{2}$$

where the modes are calculated as

$$\phi_i = \mathbf{X}\psi_i \frac{1}{\sqrt{\lambda_i}} \tag{3}$$

which is often referred to as classical POD. Since $\mathbf{X}\mathbf{X}^T \in \mathbb{R}^{n\times n}$, this can become prohibitively expensive when n is large, as is the case for flow fields. In fluid mechanics, where n is often much larger than m, the method of snapshots is used which solves the eigenvalue problem

$$\mathbf{X}^T\mathbf{X}\psi_i = \lambda_i \psi_i \tag{4}$$

which is of size $m \times m$. This formulation assumes the data to be sampled on an equidistant mesh, which is seldom true for CFD simulations. The cell volume is included in order to use POD on a non-equidistant mesh by replacing $\mathbf{X}^T\mathbf{X}$ with $\mathbf{X}^T\mathbf{W}\mathbf{X}$, where $\mathbf{W}$ is a diagonal matrix containing the cell volumes [15]. The design coefficients $a_i(d)$ are constructed as

$$a_i(t) = \sqrt{\lambda_i}\psi_i^T \tag{5}$$

In practice, this can also be solved using a Singular Value Decomposition (SVD)

$$\mathbf{X} = \mathbf{\Phi}\mathbf{\Sigma}\mathbf{\Psi}^T \tag{6}$$

with the modes $\Phi = \phi_i(\mathbf{r}) = [\phi_1, \phi_2, \ldots \phi_m] \in \mathbb{R}^{n\times m}$ and the coefficients $\mathbf{\Sigma}\mathbf{\Psi}^T = A_i(d) = [a_1, a_2, \ldots a_m] \in \mathbb{R}^{m\times m}$.

The size of the problem can be significantly reduced by truncating the number of POD-modes if a large portion of the total energy is contained in a few modes. This was done in the work by Iuliano and Quagliarella [11], where the modes were truncated to contain 95 % of the total energy. In this work, all modes are used to remove any decency on the truncation level.

2.2. Latin Hypercube sampling

The surrogate model was created from an initial number of CFD simulations in a Design Of Experiments (DOE) sampling plan. There exist several DOE techniques and, in this work, the Latin Hypercube (LHC) sampling plan was used. The LHC plan divides each design parameter into N equally sized intervals where the same value of a parameter can only occur once. An example of a randomly generated LHC plan can be seen in Fig. 1.

Randomised LHC sampling plans can suffer from clustering, leaving areas of the design space less explored. The separation between designs was increased by optimising the LHC plan using the Audze–Eglais objective function

$$\min U = \min \sum_{p=1}^{P} \sum_{q=p+1}^{P} \frac{1}{L_{pq}^2} \tag{7}$$

where L_{pq}^2 is the distance between two sample points. The optimisation of the LHC sampling plan is based on the work by Bates et al. [16], where the sampling plan is improved by minimising Eq. (7) using a permutation genetic algorithm. Fig. 2 shows the optimised sampling plan with 101 sample points that were used in this work for the two-dimensional test case. Throughout this work, an optimised LHC plan was used except for the one-dimensional case, where the samples were spaced equidistantly.

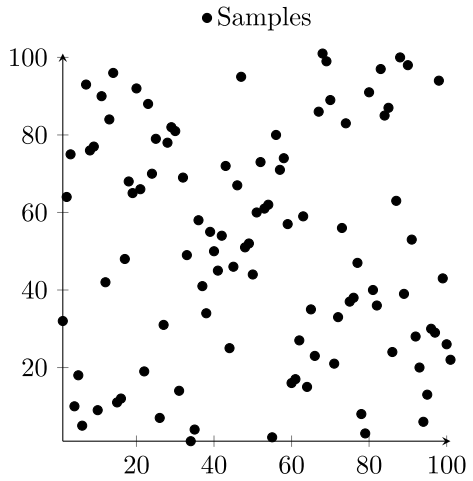


Fig. 1. Random Latin Hypercube sampling plan in two dimensions.

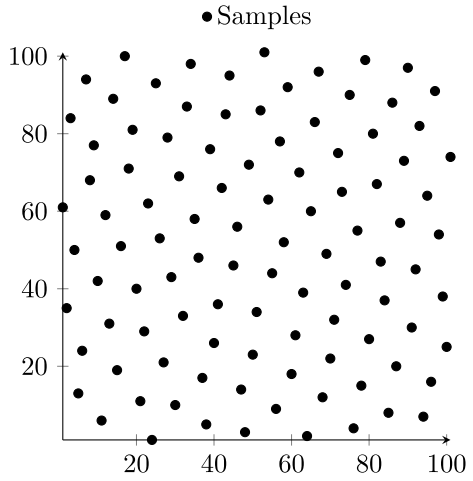
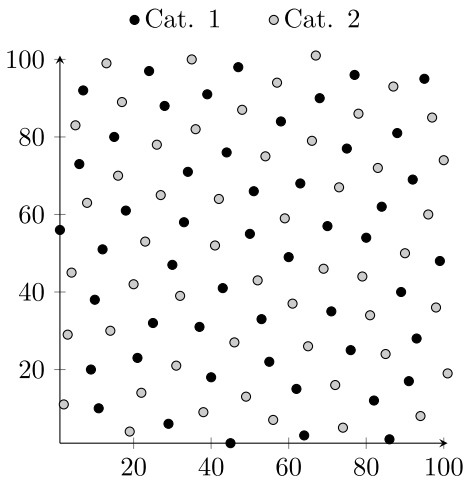


Fig. 2. Optimised Latin Hypercube sampling plan in two dimensions.

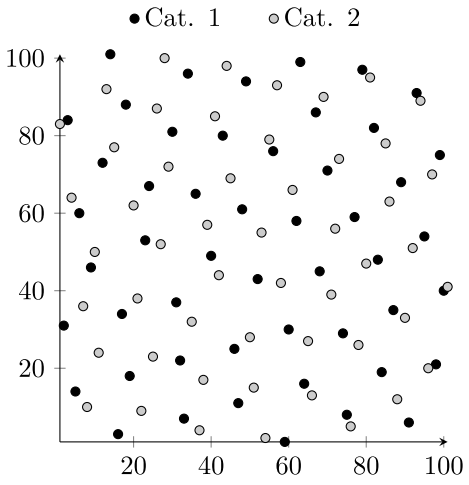
2.2.1. Categorical Latin Hypercube sampling

LHC sampling plans require an equal number of points in each dimension; this typically means that each dimension needs to be continuously variable. It can be of interest to include dimensions which are not continuously variable in the design plan. For example, the choice of vehicle tyre is not continuously variable; rather, there exists a pre-determined number of tyre options from the tyre manufacturers. To be able to mix continuous and categorical, or discrete, design parameters, a categorical dimension is added to the LHC plan where the same value is allowed to appear multiple times to fill up the categorical dimension. Note that this is no longer a strict LHC since the same design parameter appears multiple times in the categorical dimensions.

When optimising the categorical LHC, the objective function is altered to promote separation of the design points within each categorical plane. This is done by calculating the Audze–Eglais function within each category, in addition to calculating it between all points. Both objectives are then summed together using weights for each objective, where the weights are specified by the user. Promoting separation within each categorical plane ensures that the points within a categorical dimension are not clustered in one area. The effect of different weightings can be seen in Figs. 3a and 3b. Note that putting a large emphasis on the categorical separation is similar to creating two separate sampling plans. In this work, the categorical LHC uses the weights 0.9975 between all points and 0.0025 for the categorical separation.



(a) Weights Continuous = 0.9975, Categorical = 0.0025.



(b) Weights Continuous = 0.0025, Categorical = 0.9975.

Fig. 3. Categorical LHC sampling plans with varying weights. The two categories are represented by grey and black dots respectively.

2.3. Radial basis function interpolation

New flow fields are created by finding new design coefficients in Eq. (5). These are found through interpolation between existing coefficients to new, untested design locations. In this work, Radial Basis Function interpolation is used to estimate the design coefficients. The RBF interpolation is calculated as

$$u(\mathbf{x}) = \sum_{i=1}^{N} w_i \xi_i(\|\mathbf{x} - \mathbf{x_i}\|_2) \tag{8}$$

where w_i are the weights and ξ_i is the Radial Basis Function, and $\|\mathbf{x} - \mathbf{x_i}\|_2$ denotes the Euclidean distance between the new point $\mathbf{x}$ and a sample point $\mathbf{x_i}$. The RBFs, ξ_i, used in this work is the gaussian

$$\xi(r) = e^{-(\varepsilon r)^2}, \tag{9}$$

inverse quadratic

$$\xi(r) = \frac{1}{1 + (\varepsilon r)^2}, \tag{10}$$

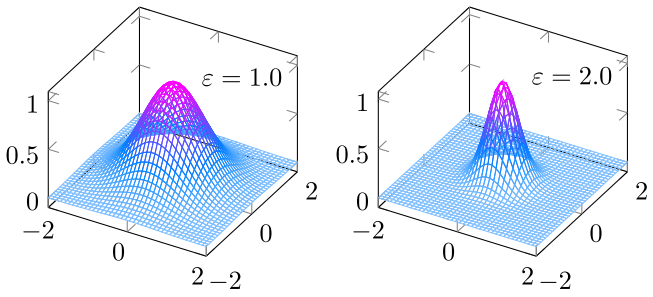


Fig. 4. Gaussian RBF with different width factors, ε.

and inverse multiquadratic

$$\xi(r) = \frac{1}{\sqrt{1 + (\varepsilon r)^2}} \tag{11}$$

basis functions where ε is the width factor determining the size of the RBF. An example of the influence of the width factor can be seen in Fig. 4.

The weights w_i are found by solving the linear system

$$\mathbf{Aw} = \mathbf{u} \tag{12}$$

where $\mathbf{A} = A_{ij} = \xi_i(\|x_i - x_j\|_2)$ and $\mathbf{u} = \mathbf{u}(\mathbf{x_i})$ are the known function values at the sample points. The chosen RBFs will result in a positive-definite matrix $\mathbb{A}$, thereby guaranteeing a unique solution to Eq. (12) [17].

2.3.1. Radial basis function hyperparameter optimisation

The tuning of the hyperparameters ξ_i and ε_i is done with the Differential Evolution algorithm de/rand/1/bin/radiuslimited with a population size of 50, crossover probability of 0.7 and differential weighting factor of 0.6. More information on Differential Evolution can be found in the book by Price et al. [18]. The objective function used to improve the interpolant is the Leave-One-Out (LOO) cross-validation error. The interpolant is created from $n-1$ points where the prediction error, e_i, for the removed point is stored. This process is repeated for all n points, and the LOO RMS error

$$E_{RMS,LOO} = \sqrt{\overline{\mathbf{e}^2}} \tag{13}$$

is used to quantify the performance of the interpolant. The LOO error for all n points is expensive to calculate, but thanks to Rippas algorithm [19], the LOO error can be estimated without computing the solution to Eq. (12) n times. The total computational cost for Eqs. (12) and (13) is on the order of $\mathcal{O}(n^4)$ without Rippas algorithm and on the order of $\mathcal{O}(n^3)$ with Rippas algorithm.

2.3.2. Dimension scaling

The design parameters used to construct the surrogate model are in the form of engineering quantities. These are typically used directly as input to the surrogate model or scaled non-dimensionless, for example from 0 to 1 as

$$x_{i,scaled} = \frac{x_i - \min(x)}{\max(x) - \min(x)}. \tag{14}$$

This can pose a problem since RBFs are radial. To illustrate this, the three-hump camel function

$$f(x, y) = 2x^2 - 1.05x^4 + \frac{x^6}{6} + xy + y^2 \tag{15}$$

is used in the interval $[-5, 5]$ for both dimensions. The function is shown in Fig. 5. When optimising the interpolant, all dimensions

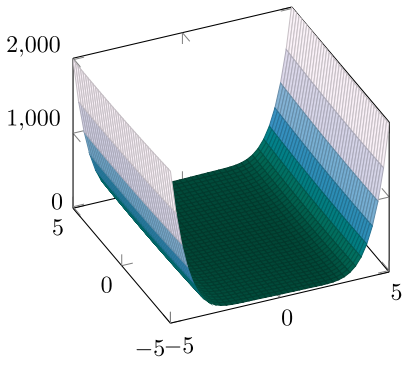
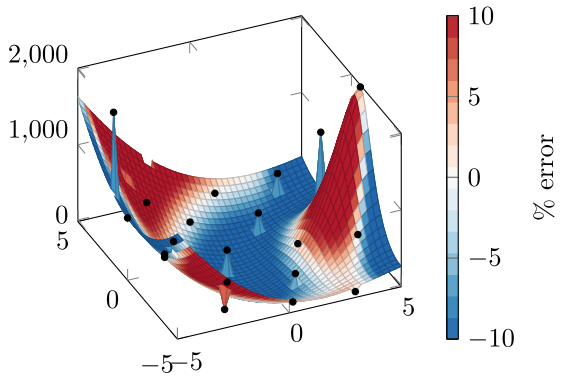
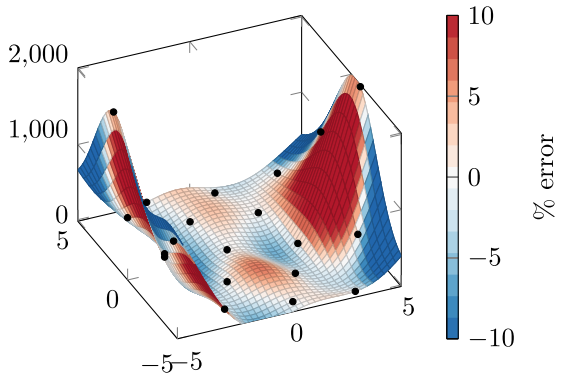


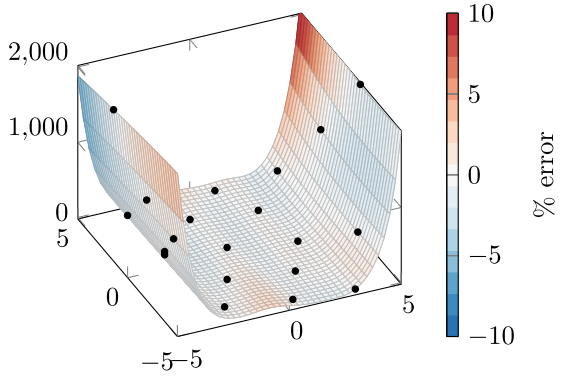
Fig. 5. Three camel hump function.



(a) Varying RBF and width.



(b) Fixed RBF and width.



(c) Adaptive scaling and fixed RBF and width.

Fig. 6. Interpolation of the three camel hump function.

are considered equally important. This leads to a compromise when selecting the RBFs, ξ_i, and the widths, ε_i. An example of this compromise can be seen in Fig. 6a.

The resulting interpolation is not representative of the overall shape of the original function. The sharp peaks in Fig. 6b occur due to the interpolants inability to fit a surface which minimises the LOO-error for all points. When it is not possible to fit a smooth surface through all points, the RBF width factor, ε_i, becomes large for some points to reduce their influence on the total LOO-error. To reduce the flexibility of the interpolants which can be fitted to a set of observations, it is possible to constrain the problem. The interpolant is constrained to use the same width and RBF for all points in Fig. 6b. While the smoothness of the function is improved, the overall fit is still poor compared to the ground truth, Fig. 5. The cause of the poor fit is the large differences in scale of each dimension when mapping from function input to output. The inputs for each design dimension need to be scaled independently to solve this. In the work by Giannakoglou [20], the design parameters are scaled based on gradient information from the response surface. In this work, the design parameters are scaled by using an additional scaling factor c_j, which is introduced to Eq. (14) as

$$x_{i,scaled} = c_j \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{16}$$

where $c_j = [c_1, c_2, \ldots, c_d]$ and d is the number of design dimensions. Note that the resulting RBFs are radial in the scaled space which, when translated back to the design space, results in the RBFs being ellipsoidal. The benefits of using adaptive scaling Radial Basis Functions on the same three camel hump function are evident when considering Fig. 6c, where a much better prediction of the global shape is obtained.

Adaptively scaling the design dimensions allows the designer to use engineering quantities for each design dimension without considering the appropriate scaling of each quantity. Furthermore, adaptively scaling each dimension reduces the penalty of including design parameters which have little or no influence on the output, i.e. constant response.

2.3.3. Kriging interpolation

Another commonly used interpolation technique, which was considered in this work, is Kriging. The authors compared the performance between Kriging and RBF for 5, 10 and 40 sample points using the one-dimensional data presented in Section 4.1.1. The RBF based interpolant outperformed Kriging over ten runs for 10 and 40 points while both methods performed similarly for 5 points. Chandrashekarappa and Duvigneau [21] compared the accuracy of Kriging- and Radial Basis Function-interpolation and found that the methods performed comparably. As concluded in the literature review by Goel et al. [5], no single surrogate performed best for all problems. It should be noted that Kriging offers the benefit of providing error estimates which can be used to explore the design space efficiently.

Adaptive scaling can be used with Kriging interpolation; however, the computational complexity of Kriging is $\mathcal{O}(n^4)$ while for RBF interpolation, it is $\mathcal{O}(n^3)$. For larger n, Kriging can become limited by the number of affordable iterations compared to RBF interpolation when optimising the scaling factor, c_j.

2.3.4. Practical considerations

The condition number of the matrix **A** in Eq. (12) can become large when optimising the RBFs and width factors. Due to finite numerical precision when solving the linear system, the condition

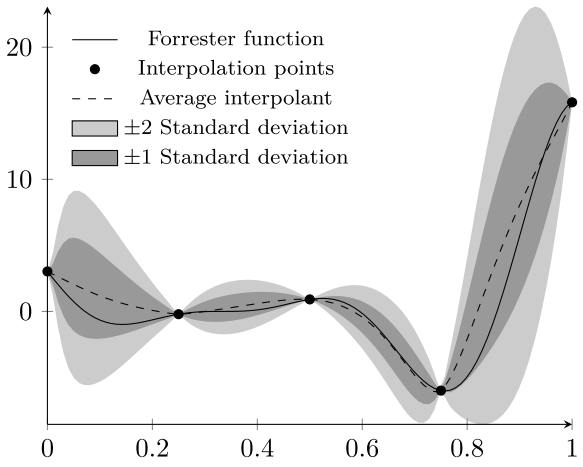


Fig. 7. Forrester function and interpolant created from five points. The standard deviation and average based on 1000 runs.

number of matrix **A** needs to be considered to keep the desired number of significant digits. Similarly, the numerical precision of the dimension scaling needs to be considered when scaling between small and large numbers.

When dealing with data containing noise, the RBF interpolation exhibits high-frequency oscillations when the number of interpolation points increases. Which is due to the interpolant creating a fit containing the noise as well as the global function shape. Fasshauer [22] suggests using ridge regression when dealing with data containing noise. A regression term, λ, is added to the diagonal of **A**, Eq. (12). This term relaxes the requirements of the surrogate model being directly interpolating, which reduces high-frequency oscillations in the surrogate model when modelling a problem with noise. The ridge regression term is treated as an additional hyperparameter and is optimised together with the other RBF hyperparameters.

The presented methodology is stochastic, and therefore, does not yield the same solution each time. To illustrate this, an interpolant based on the Forrester function

$$f(x) = (6x - 2)^2 \sin(12x - 4), \tag{17}$$

is recreated 1000 times based on five equidistantly spaced sample points.

As mentioned, the interpolants hyperparameters are optimised using DE where the LOO-error is used as the objective function to tune the RBFs, ξ_i, and the widths, ε_i. In this example, the optimisation of the interpolant is stopped after 1×10^4 function calls. The average interpolant, along with the standard deviation based on the 1000 interpolants, can be seen in Fig. 7. Optimising the width factor and RBF for each point allows for flexibility when fitting the interpolant to the sample points. However, this flexibility might lead to overfitting, where the number of free parameters is large compared to the problem [23]. A demonstration of the possible effects of a large number of free parameters is given by Mayer et al. [24]. Reducing the degrees of freedom by using the same width and RBF for all points can alleviate the problem of overfitting; however, it is not clear which degree of freedom is suitable when constructing the interpolant. The impact of the degrees of freedom on the resulting surrogate model will be investigated in the results.

In the work by Abedinia and Amjady [25], the trade-off between under- and over-fitting was improved by optimising the number of RBF centres and their placement. Srivastava et al. [26] proposed a method where the neurons of a neural net are randomly turned off at training time. This helps regularise the resulting network to act like an average of an ensemble of networks.

2.4. Surrogate model

As mentioned, new flow fields are created by interpolating the coefficients in Eq. (5). The prediction accuracy of each new design is evaluated using the integration planes in Fig. 11. The entire procedure is outlined in Algorithm 1.

Algorithm 1: POD-based optimisation.

Result: Optimised geometry
Create optimised Latin Hypercube Sampling plan;
Simulate all the designs in the plan;
Interpolate $\mathbf{X}(\mathbf{r}, d)$ to common mesh;
while *within computational budget* **do**
  $\Phi_{n\times m}, A_{m\times m} = \mathbf{POD}(\mathbf{X})$;
  foreach *row in A* **do**
    $\mathbf{RBFinterp} = \mathbf{optimiseRBFinterp}(row)$;
  end
  while *Evolutionary Algorithm* $\neq$ *Converged* **do**
    for $i \leftarrow 1$ **to** m **do**
      $a_{i,new} = \mathbf{RBFinterp}_i(d_{new})$
    end
    $\mathbf{X_{new}}(\mathbf{r}, d_{new}) = \sum_{i=1}^{m} a_i(d_{new})\phi_i(\mathbf{r})$;
    $objective = \mathbf{integrationPlanes}(\mathbf{X_{new}}(\mathbf{r}, d_{new}))$;
  end
  Add design(s) found by EA to sampling plan;
  Simulate the new designs in the sampling plan;
  Interpolate $\mathbf{X}(\mathbf{r}, d)$ to common mesh;
end

The POD-based approach is compared to the force-based method, where the objective value is used directly to create the surrogate. Using the objective value to create the surrogate model is often termed Response Surface Modelling (RSM) while the POD-based method is commonly termed reduced-order Modelling (ROM). The asymptotic cost for the POD-based method is $\mathcal{O}(n^4)$, while being $\mathcal{O}(n^3)$ for the force-based method. The complexity for creating one surrogate model is dominated by the cost of solving the linear equation system, equation (12), which is an $\mathcal{O}(n^3)$ operation. The POD-based cost of $\mathcal{O}(n^4)$ is due to the number of interpolants needed for the POD-based method is equal to the number of modes which is equal to the number of samples. For the force-based method, the number of interpolants needed is equal to the number of objectives. The computational cost of the POD-based method can be reduced by using a truncated set of modes so that the cost final cost scales with $\mathcal{O}(n^3)$; however, in this work, all POD modes are retained. In practice, the POD-based method will be the number of modes, m, times more expensive than the force-based method. The storage cost is another factor to consider when using the POD-based method as the flow field needs to be saved for each design, whereas for the force-based method, the storage is negligible.

2.5. Software

The CFD simulations were performed in the commercial flow solver Star-CCM+. All the surrogate modelling and design plan construction was performed in the Julia programming language [27] using the packages LatinHypercubeSampling.jl [28] for the sampling plans and ProperOrthogonalDecomposition.jl [29] for the POD construction. The RBF interpolation was done with ScatteredInterpolation.jl [30] and the Differential Evolution algorithm was provided by BlackBoxOptim.jl [31].

The packages to perform the optimisation are bundled into one package, SurrogateModelOptim.jl [32], with functionality to

Fig. 8. DrivAer in the original estate configuration. 5.5° roof angle and 87.5 mm diffuser height.

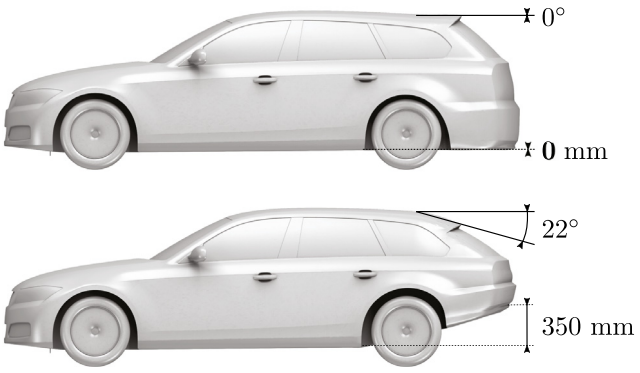


Fig. 9. Roof angle and diffuser height design parameters.

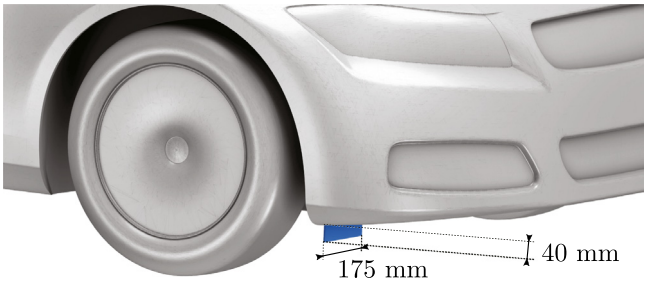


Fig. 10. Front-wheel deflector size.

perform the entire optimisation routine for convenience. All benchmark results for the surrogate model benchmark can be found in [32] as well as the script used to run the benchmark.

3. Test cases

3.1. Aerodynamic surrogate accuracy test

The surrogate model accuracy was tested on the generic vehicle model DrivAer [12], developed by the Technical University of Munich, in the estate configuration. The variant, shown in Fig. 8, features a smooth underbody, closed rims and no cooling flow. The method performance is investigated using three design parameters, for one dimension, roof angle; two dimensions, roof and diffuser angle; and three dimensions, roof angle, diffuser height and front-wheel deflector on/off. The roof angle and diffuser height extreme points can be seen in Fig. 9. The front-wheel deflector is 175 mm wide and 40 mm tall and is located 100 mm inboard of the front-wheels outer face and 85 mm upstream of the front-wheels most forward point, see Fig. 10. Each simulation was run steady-state for half the vehicle model and a symmetry plane boundary condition was used to account for the other half of the vehicle. The meshes consisted of approximately 22×10^6 cells and the realisable $k-\varepsilon$ RANS turbulence model was used.

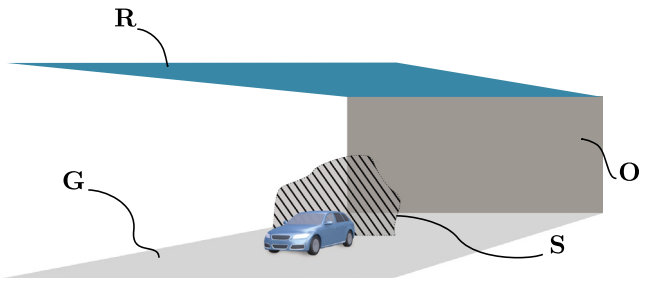


Fig. 11. Planes used to evaluate lift and drag. Wake plane S extends to the domain edges. O, R and G are the domains outlet, roof and ground respectively.

Each simulation was run for at least 2000 iterations or until the drag coefficient varied less than $\pm 0.001\,C_D$ for the last 500 iterations to ensure the solution was stable.

Ridge regression was not used for the aerodynamic accuracy comparison to retain an exactly interpolating surrogate model.

3.1.1. Mesh

Proper Orthogonal Decomposition requires the dataset to be ordered consistently between snapshots. This is typically achieved through volume mesh deformation [10]. In this work, each CFD simulation is run on topologically different, unstructured, meshes and is later interpolated to a common mesh. The common mesh does not include the vehicle body since it changes between each design; however, the wheel geometry is constant and kept in the common mesh. The results are interpolated to the common mesh using a weighted nearest neighbour interpolation where the distances to the nearest neighbour and its immediate neighbours in the original mesh are used to compute the weights.

Since each simulation is interpolated to a mesh without a vehicle body, it is no longer possible to use the pressure and shear forces acting on the vehicle surface to compute the drag and lift. To solve this, the drag and lift area, C_DA and C_LA respectively, are evaluated in the far-field as

$$C_DA = \int_S -C_P - 2\left(\frac{V_x^2}{V_\infty^2} - \frac{V_x}{V_\infty}\right) \mathrm{d}S \tag{18}$$

$$C_LA = -\int_O \frac{2V_xV_z}{V_\infty^2}\mathrm{d}O - \int_R C_P\mathrm{d}R + \int_G C_P\mathrm{d}G \tag{19}$$

where the integration planes, S, O, R and G, are defined in Fig. 11.

The mesh interpolation and subsequent evaluation of forces in the wake introduce errors in the force calculation. Based on 101 simulations of the DrivAer with varying roof angle, the errors were $(0.005 \pm 0.002)\,C_D$ and $(0.000 \pm 0.004)\,C_L$. The constant offset of $0.005\,C_D$ is due to the vehicle and ground interaction. This can be accounted for by including the drag on the ground plane located between the inlet and plane S. Due to the offset being near-constant, the ground plane drag was not considered in this work.

3.2. Benchmark test cases

The surrogate model performance, when used for optimisation, was tested with nine benchmarking functions both with and without noise resulting in a total of 18 cases. Each benchmark was started from five sample points in an optimised LHC and run for 95 iterations where the surrogate model was recreated between each iteration. The small initial sample size was chosen to verify the methods convergence capability when starting with a sparse sampling plan.

When performing aerodynamic vehicle optimisation in a wind tunnel or using numerical simulations, the results from each test

Table 1
Benchmark test functions.

Problem	Definition
f_1 Styblinski–Tang 2D	$\frac{1}{2}\sum_{i=2}^{2}\left(x_i^4 - 16x_i^2 + 5x_i\right)$
f_2 Rastrigin 2D	$20 + \sum_{i=1}^{2}\left[x_i^2 - 10\cos(2\pi x_i)\right]$
f_3 Rosenbrock 2D	$\sum_{i=1}^{2}\left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$
f_4 Beale 2D	$(1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$
f_5 Sphere 2D	$\sum_{i=1}^{2} x_i^2$
f_6 Perm d, β 2D	$\sum_{i=1}^{2}\left(\sum_{j=1}^{2}(j^i + \beta)\left(\left(\frac{x_j}{j}\right)^i - 1\right)\right)^2$
f_7 Goldstein–Price 2D	$[1 + (x_1 + x_2 + 1)^2(1914x_1 + 3x_1^2 14x_2 + 6x_1x_2 + 3x_2^2)] * [30 + (2x_1 3x_2)^2(1832x_1 + 12x_1^2 + 4x_2 36x_1x_2 + 27x_2^2)]$
f_8 Hartmann 6D	$-\sum_{i=1}^{4}\alpha_i \exp\left(-\sum_{i=1}^{6}A_{ij}(x_j - P_{ij})^2\right)$ $\alpha = \begin{pmatrix}1.0 & 1.2 & 3.0 & 3.2\end{pmatrix}$ $\mathbf{A} = \begin{pmatrix}10 & 3 & 17 & 3.5 & 1.7 & 8\\ 0.05 & 10 & 17 & 0.1 & 8 & 14\\ 3 & 3.5 & 1.7 & 10 & 17 & 8\\ 17 & 8 & 0.05 & 10 & 0.1 & 14\end{pmatrix}$ $\mathbf{P} = 10^{-4}\begin{pmatrix}1312 & 1696 & 5569 & 124 & 8283 & 5886\\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991\\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650\\ 4047 & 8828 & 8732 & 5743 & 1091 & 381\end{pmatrix}$
f_9 Rosenbrock 12D	$\sum_{i=1}^{12}\left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$

contain some level of noise. The repeatability, when performing wind tunnel tests, without removing the vehicle, is on the order of $\pm 0.001\,C_D$ [33] and typical changes to the vehicle drag vary from $0.010\,C_D$ to $0.050\,C_D$ depending on the development stage. Interpreting the repeatability as noise, this translates to a noise level of 2 % to 10 %. The optimisation methods are benchmarked with 10 % noise to cover noise levels seen in aerodynamic vehicle optimisation. The noise level is determined by the maximum and minimum function value and the noise is uniformly distributed pseudorandom white noise. The ridge regression coefficient λ was treated as a hyperparameter for both the smooth and noisy test cases as to not make any underlying assumption of the noise level to verify the methods ability to interpret the noise level automatically.

The nine benchmark functions are available in Table 1. Both multimodal and unimodal benchmark problems were selected as well as multidimensional problems to verify the surrogate models ability to handle cases similar to those seen in aerodynamic optimisation. Note that the output of these functions is a single scalar, and it is only the force-based method which is benchmarked.

Each function was evaluated in the search space presented in Table 2. The function values for each benchmark problem are scaled from 0 to 1 based on the global maximum and minimum function value.

The results of each benchmark problem are presented in comparison with four other gradient-free optimisation algorithms, Random Sample (RS), Differential Evolution (DE), Nelder–Mead (NM) and Bayesian Optimisation (BO). Random Sample is the conceptually simplest of the methods and works by randomly sampling the search space and returning the solution with the

Table 2
Benchmark test function search space.

Problem	Search space
f_1 Styblinski–Tang 2D	$x_i \in (-5.0, 5.0)$ for all i
f_2 Rastrigin 2D	$x_i \in (-5.12, 5.12)$ for all i
f_3 Rosenbrock 2D	$x_i \in (-5.0, 5.0)$ for all i
f_4 Beale 2D	$x_i \in (-4.5, 4.5)$ for all i
f_5 Sphere 2D	$x_i \in (-5.12, 5.12)$ for all i
f_6 Perm d, β 2D	$x_i \in (-2.0, 2.0)$ for all i
f_7 Goldstein–Price 2D	$x_i \in (-2.0, 2.0)$ for all i
f_8 Hartmann 6D	$x_i \in (0.0, 1.0)$ for all i
f_9 Rosenbrock 12D	$x_i \in (-5.0, 5.0)$ for all i

currently found best fitness after each iteration. The Nelder–Mead, or simplex search, algorithm constructs a simplex from $d+1$ function calls, the simplex is then either reflected, expanded, contracted, or shrunk. The Nelder–Mead optimisation algorithm was supplied by Optim.jl [34]. More information about the simplex algorithm can be found in the works by Nelder and Mead [35] and Gao and Han [36]. The parameters used in the NM algorithm are those suggested by Gao and Han [36]

$$\alpha = 1, \quad \beta = 1 + 2/d, \quad \gamma = 0.75 + 1/2d, \quad \delta = 1 - 1/d.$$

Bayesian Optimisation is suited for expensive functions and constructs a surrogate model of the problem and quantifies the underlying uncertainty using Gaussian process regression. The Bayesian optimisation algorithm, called Dragonfly, was supplied by Kandasamy et al. [37]. More information on Bayesian Optimisation can be found in the work by Frazier [38].

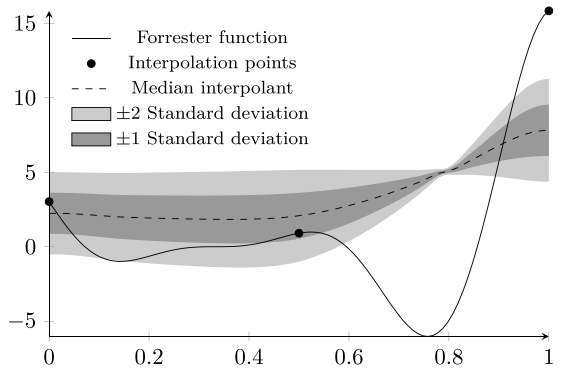
The parameters used in the DE algorithm are the same as used for optimising the RBF hyperparameters except for the population size, which was reduced to 10, to balance the trade-off between population size and function evaluations. Note that this population size is small; however, the total number of function evaluations in this comparison was also small. Populations sizes of 5 and 50 were also tested; however, both performed worse compared to a population size of 10.

The surrogate model hyperparameters were optimised for 5000 iterations where the first 2500 iterations were optimised with an additional constraint of using the same RBF and width. Optimising the hyperparameters with the constraint improved the accuracy of the surrogate model as suitable dimension scaling, and ridge regression factors are found faster. This is especially true when the number of samples is large. The LHC sampling plan was optimised for 1000 iterations.

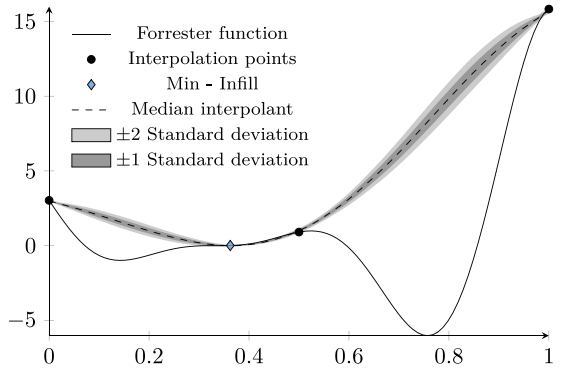
3.2.1. Infill

The surrogate model was used to determine the next sample location after each function evaluation. Infill strategies which explore the design space help the optimisation process to stay on target, finding the global optimum, without getting stuck in local minima. Exploitive strategies pick the minimum location predicted by the surrogate model. There also exist variations which try to take advantage of both, such as Expected Improvement which is commonly used with Bayesian optimisation. In this work, an ensemble of 10 surrogates was created and used to determine new infill locations, inspired by Goel et al. [5]. The surrogate prediction was calculated from the median of the 10 surrogate models.

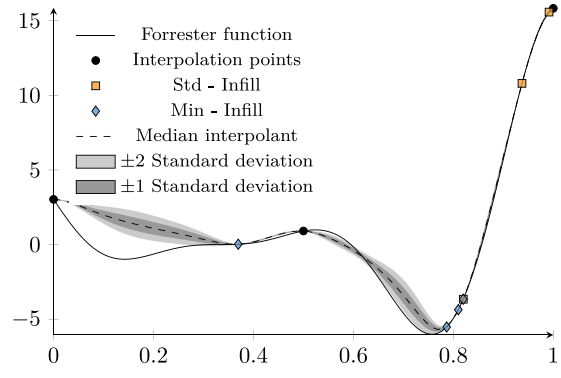
A mix of two infill criteria was used to determine new sampling locations in the surrogate model, minimum prediction and maximum standard deviation of the surrogate models. An example of optimising the Forrester function, Eq. (17), can be seen in Fig. 12. The first surrogate model estimate is a poor representation of the overall function due to the small number of sample points. Notice that the first surrogate model prediction



(a) Three sampling points.



(b) Three sampling points and one minimum infill point.



(c) Three sampling points and seven infill points.

Fig. 12. Forrester function optimisation.

is not interpolating, i.e. it is not able to fit a good RBF interpolant without increasing the ridge regression factor. As the number of sample points increases, the function is correctly identified as noise-free.

It is believed that the algorithm will converge to the global optimum as the number of points tend to infinity due to the sampling of the standard deviation. The standard deviation is low close to sampled areas which will increase the coverage in space as samples are added. However, the intent is to use the surrogate model with expensive functions where the rate of improvement is the most important aspect of the optimisation algorithm.

Table 3
Surrogate performance configurations.

Case	RBF, ξ_i & Width, ε_i	Scaling c_j
A	Constrained	None
B	Variable	None
C	Constrained	Adaptive
D	Variable	Adaptive

In the example, the criteria were alternated between pure exploitation, using the median prediction, and pure exploration, using the standard deviation of the surrogate ensemble. For the benchmark tests, every odd sample was selected based on pure exploitation and every even sample by a weighted sum of the surrogates median and standard deviation as

$$\min \quad w \cdot median(x) + (1 - w) \cdot -1 \cdot std(x).$$

A weight of 0.0 results in an explorative search, while a weight of 1.0 is purely exploitative. For every even sample, the weight was cycled through 0.0, 0.3, 0.6, 0.9. The performance was increased by using the weighted formulation compared to using pure exploration for every even sample. This is due to more samples being spent in regions of predicted low function value, particularly when the number of samples increase and the standard deviation in the ensemble reduces.

The infill objective is optimised each iteration using the previously presented DE algorithm for 25 000 iterations. The package bundle SurrogateModelOptim.jl [32] includes convenience functions to create an optimised surrogate model. This is useful if the user wants to try other metaheuristic optimisation algorithms such as ant colony- or particle swarm-optimisation for example, to find promising new infill locations.

4. Results & discussion

4.1. Aerodynamic surrogate accuracy-test

The accuracy of the interpolants is investigated in four cases, listed in Table 3. The constrained RBF and width refers to the use of the same RBF and width factor for all points, while for the variable RBF and width, is optimised per RBF centre, which in this case is per sample point. Note that the interpolant is optimised in each case. For cases A and B, the engineering quantity, such as measured distance or angle, is used directly as input to the surrogate while for cases C and D, the engineering quantity is adaptively scaled. This is done for both the POD-based and force-based surrogate model.

The results are presented for drag only since lift follows a similar trend. The results are presented successively for one dimension, two dimensions and three dimensions. Each dimension investigation is performed by running a DOE of 101 CFD simulations, where a subset of the simulations are used to create the surrogate model, and the remaining points are used as a validation set. The performance indicator used is the Mean Absolute Error (MAE)

$$MAE = \frac{\sum_{i=1}^{n} |f_i - \tilde{f}_i|}{n} \tag{20}$$

ith CFD result, or ground truth, and $\tilde{f}_i$ is the ith surrogate prediction. The MAE performance indicator was used as it does not bias the analysis toward outliers like other indicators such as Root Mean Squared Error does. This indicator was chosen since the surrogate model is recreated several times when used for optimisation and it is thus more important to have good performance on average. In the case of the two- and three-dimensional test cases, the sample points are chosen as a subset

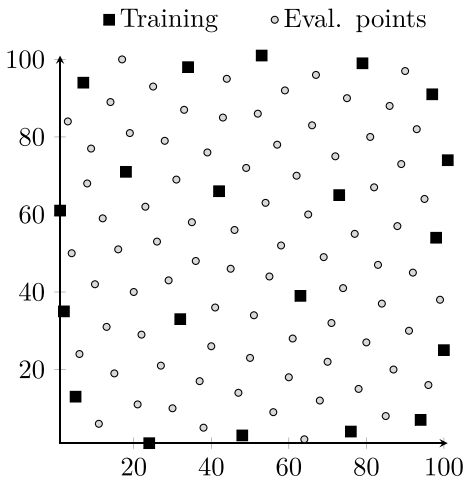


Fig. 13. Optimised subset LHC plan in two dimensions for a subset of 20 points, also seen are the points used to evaluate the surrogate performance.

of an LHC plan containing all 101 simulations. The subset is chosen by optimising the Audze–Eglais error as a subset from the original plan. Note that extracting a subset from an existing plan will result in an Audze–Eglais objective which is worse than creating an entirely new plan; however, this significantly reduces the computational expense since one dataset can be used to investigate the performance for several subsets. Creating the surrogate from an optimal subset of the entire optimised LHC plan ensures separation between the evaluation points and the points from which the surrogate is created, which can be seen in Fig. 13. The performance of the surrogates is investigated in subsets of 10, 20 and 40 points for each case and for one, two and three design dimensions.

Due to the method being stochastic, each interpolant is created 25 times. The presented performance data indicates the median performance with ±95% confidence interval for the median unless stated otherwise. The median confidence interval for the lower interval is calculated as

$$\left\lfloor \frac{1}{2}\left(n - 1.96\sqrt{n}\right) \right\rceil \tag{21}$$

and for the upper interval is calculated as

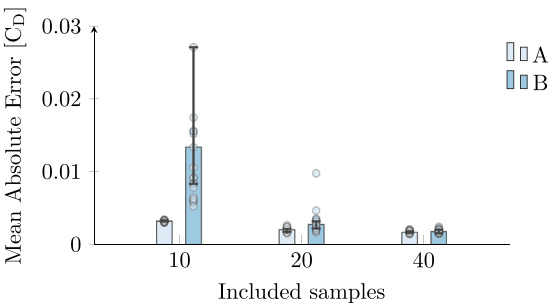
$$\left\lfloor \frac{1}{2}\left(2 + n + 1.96\sqrt{n}\right) \right\rceil \tag{22}$$

where $\lfloor x \rceil$ denotes rounding to the nearest integer. The samples from which the median is created are shown; however, for illustration purposes, the y-axis is limited with some outliers not visible. An example can be seen in Fig. 14a.

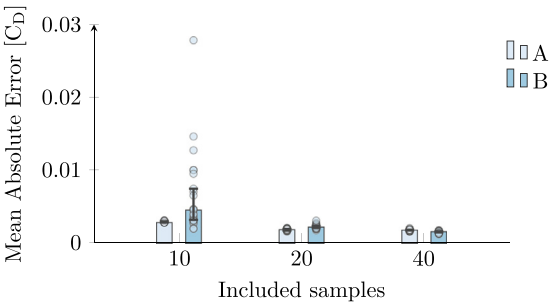
4.1.1. One dimension

The roof angle is used to investigate the surrogate model performance for one dimension. In one dimension no relative scaling is used, i.e. it is only cases *A* and *B* which are investigated. The design parameter is varied from 0° to 22° in steps of 0.22°, resulting in 101 simulations. The maximum/minimum drag and lift coefficient varied from 0.231 C_D to 0.272 C_D and −0.208 C_L to 0.128 C_L for the investigated samples. It should be noted that the investigated range for the design parameter is larger than what is typically considered when working with external vehicle aerodynamics.

The method performance for the POD-based and force-based surrogate can be seen in Figs. 14a and 14b. Both surrogate models perform similarly for 20 and 40 samples while the POD-method performs worse for 10 sample points.



(a) POD-based surrogate performance.



(b) Force-based surrogate performance.

Fig. 14. 1D, surrogate performance. Bar of the median performance, the error bars indicate the ±95% confidence interval for the median. The samples are shown as circles.

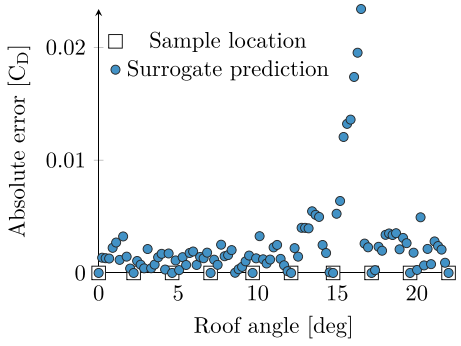


Fig. 15. Performance of the median force-based surrogate for 10 sample points, case C.

It is interesting to note that the spread between samples is larger for the POD-based surrogate. Case B, being a surrogate model created using different RBFs and widths, has a larger spread between values compared to case A, particularly for 10 sample points. This can be explained by the degrees of freedom being too large for the number of sample points, leading to overfitting.

The surrogate performance increases as the number of points increase; however, the improvement is gradual. The prediction error for each sample is investigated further for the surrogate with median performance, Fig. 15. The performance is largely within 0.005 C_D absolute error except for around 16° roof angle, where the error is large. Fig. 16 shows all 101 CFD simulations as well as the sample locations for the 10 samples. At around 16°, the simulation results are discontinuous. This is due to the flow separating from the roof, causing a sudden increase in drag. It is

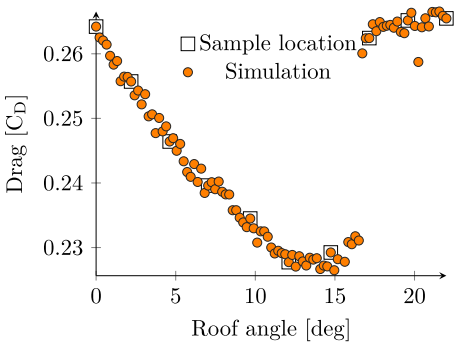


Fig. 16. Simulated drag values of all 101 simulations for varying roof angle.

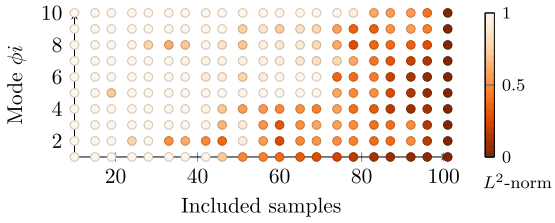


Fig. 17. Convergence history of first 10 POD modes.

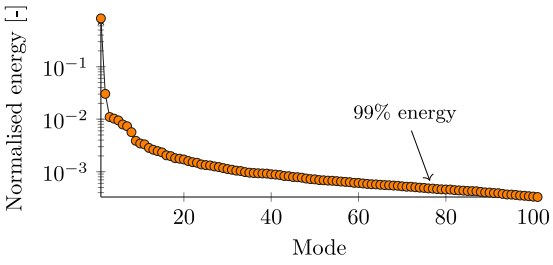
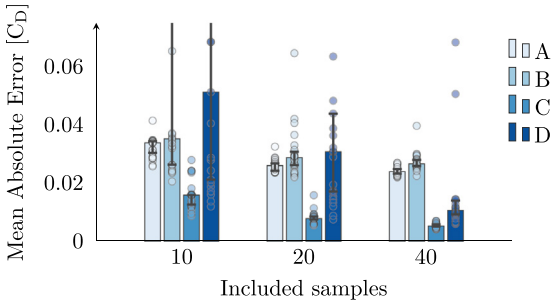
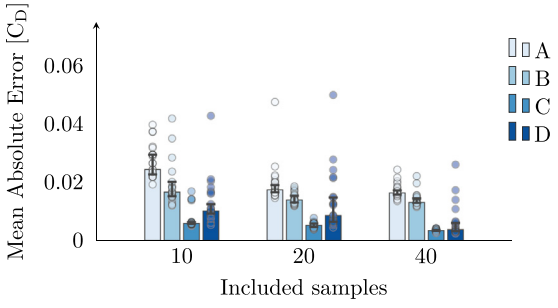


Fig. 18. Normalised energy per POD mode.



(a) POD-based surrogate performance.



(b) Force-based surrogate performance.

Fig. 19. 2D, surrogate performance. Bar of the median performance, error bars indicate the ±95% confidence interval for the median. The samples are shown as circles.

believed that the limited increase in surrogate performance, as the number of samples increase, is due to the discontinuity in drag. It is a known fact that the performance of RBF interpolation suffers in the neighbourhood of discontinuity or strong gradients [17,39]. Jakobsson et al. [17] developed the rational Radial Basis Function interpolation, which better approximates steep gradients; however, rational RBFs were not tested in this work. The samples after separation are in a design region which is not desirable, removing these samples and containing the problem could improve the surrogate performance. Additionally, weighting the LOO-error to favour the surrogate performance of points with low drag could improve the interpolant near the minimum.

The POD-based method's relatively poor performance at 10 sample points is investigated further by looking at the convergence of POD modes. The mode convergence can be used to determine if additional entries to the snapshot matrix **X** would increase the information gained. Muld et al. [15] used two metrics to determine if the POD modes were converged, the orthogonality between modes and the L^2-norm. It was concluded that both methods give good results, here the L^2-norm is used, defined as

$$L^2 = \min\left(\|\phi_{i,d} - \phi_{i,end}\|, \|\phi_{i,d} - (-\phi_{i,end})\|\right) \tag{23}$$

for the ith modes. Each mode is normalised to length 1 before the comparison. The mode $\phi_{i,end}$ is the ith mode used for comparison, containing the snapshots for all 101 designs. The minimum is used since the sign of each mode is not known. If the L^2-norm approaches 0 before the last snapshot it is considered converged, that is, adding additional snapshots will not add more information. The snapshot convergence for the first 10 modes is presented in Fig. 17. The first POD mode shows signs of convergence; however, the remaining 9 modes do not. This is thought to be the reason why the POD-based surrogate performance is particularly poor if the number of samples is small.

The energy contained in each mode indicates how many modes are needed to capture the variance in the dataset. Fig. 18 shows the energy contained in each POD mode based on all 101 simulations. The first mode contains a large portion of the overall energy (83%); however, 77 modes are needed until 99% of the energy is captured. Since a large number of snapshots are needed, it indicates that there is not much underlying generality in the dataset which can be extracted using POD. It is a known fact that POD is not capable of decomposing travelling wave problems which might explain the slow convergence and the large number of POD modes needed to reconstruct the dataset [14].

4.1.2. Two dimensions

The two-dimensional design parameters are the roof angle and the diffuser height. The roof angle is varied from 0° to 22° and the diffuser height from 0 mm to 350 mm, with 0 mm being defined as a flat floor, Fig. 9. Based on the LHC plan with 101 simulations, the drag varied from 0.227 C_D to 0.284 C_D and the lift from −0.214 C_L to 0.211 C_L.

The performance for cases A through D can be seen in Fig. 19. The performance difference between the POD- and force-based methods is larger for the two-dimensional case compared to the one-dimensional test, with the force-based method being the more accurate surrogate model.

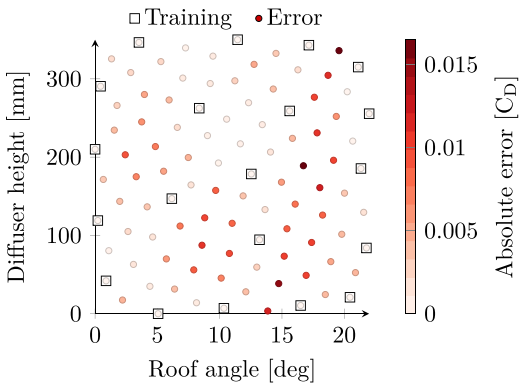


Fig. 20. Performance of the median force-based surrogate for 20 sample points, case C.

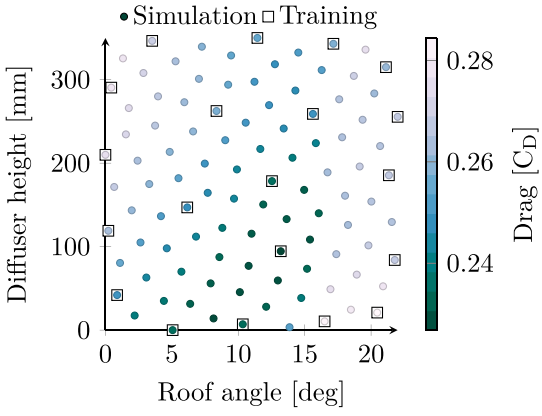
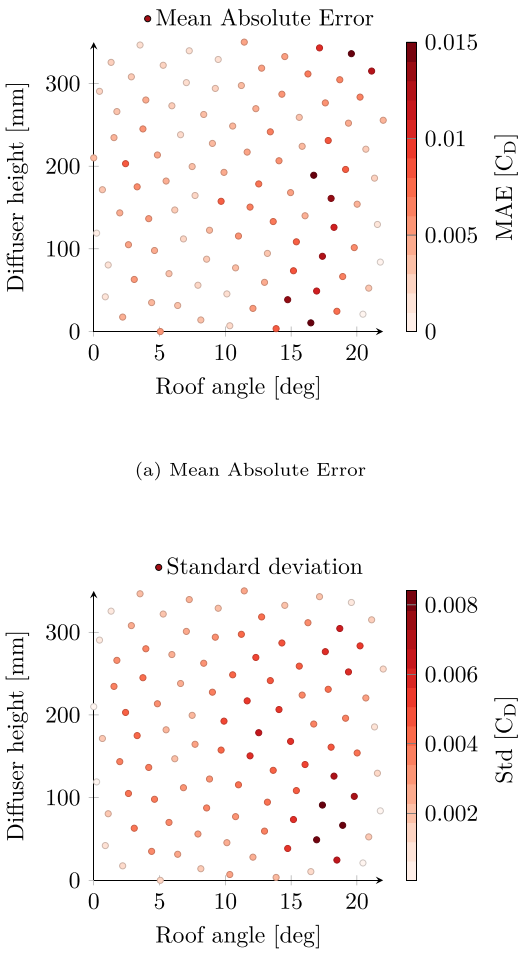


Fig. 21. Simulated drag values for all 101 simulations. Sample points of the median force-based surrogate for 20 sample points, case C.



(a) Mean Absolute Error

(b) Standard deviation

Fig. 22. Mean absolute error and standard deviation of 25 surrogates created from 20 sample points, case C.

It should be noted that the confidence interval for the median is generally larger for the POD-based method due to the larger spread. Even though the confidence interval of the median is large, the force-based surrogate consistently outperformed the POD-based surrogate.

The best performing interpolant is case C, i.e. using the same RBF and width factor with adaptive scaling. The adaptive scaling results in better performance overall for both surrogate models. The surrogate performance for cases A and C, which use the same RBF and width factor, generally results in less spread between the surrogates' performance, compared to B and D. The median performing force-based surrogate is investigated to increase the understanding of the surrogate model's shortcomings. The absolute error for the median surrogate and the associated training locations can be seen in Fig. 20 with the associated drag values in Fig. 21. At around 16° roof angle the flow separates in the two-dimensional case, similar to the one-dimensional case; however, there is an interaction between the diffuser height and angle at which the roof flow separates since the diffuser angle influences the separation point. In the region of large differences in drag, the predictive performance of the surrogates tends to decrease.

Due to the stochastic nature of the surrogate model construction, the Mean Absolute Error for the 25 surrogates is investigated in Fig. 22a. Note that the error for all points is shown since the subset is stochastic, resulting in different subsets being used for the surrogate construction.

The region of largest MAE is located close to the point of separation, at around 16° roof angle, which is an expected result.

As mentioned in the introduction, Goel et al. [5] used an ensemble of surrogates as an indicator of the local error. This indicator can be used to add infill points to the surrogate to increase its accuracy. The use of the standard deviation as an error indicator is also investigated here. The standard deviation for predicted drag for the 25 surrogates can be seen in Fig. 22b.

Comparing Figs. 22a and 22b, the areas of high standard deviation are located close to regions where the MAE is larger, around 16 deg roof angle. Note that the standard deviation for the surrogate is only shown at the sample locations since this is the only locations which can be used to verify the local error. It is, of course, possible to determine the standard deviation of surrogates at any location in the design space. This is a promising result for the use of the standard deviation of several surrogates as an indicator of potential infill locations.

4.1.3. Three dimensions

The third design parameter added to the test case is the front-wheel deflector on/off. The influence of the front-wheel deflector was tested for one of the sample points. The tested configuration has a roof angle of 4.6 deg and a diffuser height of 74 mm. The front-wheel deflector reduced drag by 0.009 C_D and reduced lift by 0.007 C_L for this configuration.

The performance for each case, number of sample points, and the comparison between the POD-based and force-based surrogate are similar to the two-dimensional test case and therefore

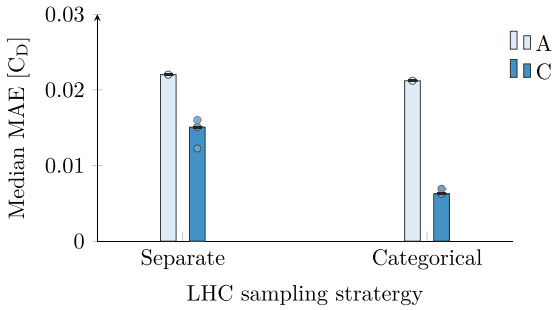


Fig. 23. Separate LHC plans compared with one categorical plan for cases A and C using 20 sample points in total.

are not shown here. The results for three design parameters show an overall larger error, as expected.

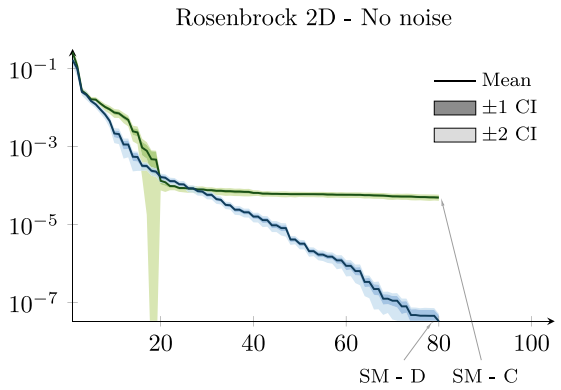
In this work, the Latin Hypercube Sampling plan was extended to include categorical, or discrete, design parameters. The performance difference between using separate LHC plans per category or using one categorical LHC is presented here. In this comparison, a subset of 20 sample points is used. Out of these 20 points, half is used to create a surrogate for front-wheel deflector on, while to other half is used to create a surrogate for front-wheel deflector off. These two separate surrogates are compared with a surrogate using all 20 points. This process is repeated 25 times for cases A and C to gather statistics due to the stochastic nature of the surrogate creation.

The results can be seen in Fig. 23, where the performance for the unscaled surrogate model, case A, shows a small benefit of using the categorical LHC plan. The performance when using adaptive scaling improves for the separate LHC plans as seen previously; however, when building the surrogate from the categorical LHC plan, the overall predictive capability of the surrogate model is greatly improved. Note that the error bars for the median are small and not visible in Fig. 23. The negligible performance gain for case A when using the categorical LHC is thought to be limited by the compromise of choosing RBF and width factor when included design parameters are of largely differing scales. This is similar to the interpolation performance of the three camel hump function presented in Section 2.3.

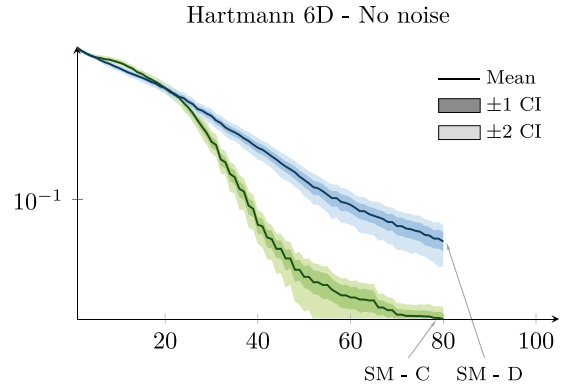
4.2. Benchmark test cases

Based on the accuracy of the aerodynamic test case, the surrogate models C and D, which both use adaptive dimensional scaling, are selected for further benchmarking. Surrogate models C and D are compared with test functions f_3 and f_8, Rosenbrock 2D and Hartmann 6D, in Fig. 24. The increased flexibility of the surrogate model D in the two-dimensional case increases the convergence speed after approximately 20 function evaluations. The six-dimensional test function indicates slower convergence when using surrogate model D; this is thought to be due to overfitting when the number of free parameters is too large compared to the problem being solved. It is not clear when there are enough samples for surrogate model D to outperform C. The benchmark results are only presented for surrogate model D from here on.

Each benchmark function was run 100 times for the surrogate model to gather statistics where the confidence intervals are calculated using bootstrapping. The Bayesian Optimisation algorithm was run 20 times for each function due to the high computational cost. The other optimisation algorithms are run a minimum of 100 times; however, sometimes more runs were required to reduce the confidence intervals and give meaningful comparisons of the mean performance.



(a) Rosenbrock 2D benchmark function.



(b) Hartmann 6D benchmark function.

Fig. 24. Surrogate model optimisation performance on two test functions using surrogate model versions C and D.

Fig. 25 shows the performance history for all nine benchmark functions without noise for a maximum of 500 function calls. The surrogate model outperforms the other methods in eight out of the nine tested functions.

The surrogate model shows good initial convergence speed which is important when optimising with a fixed time budget. It is important to note that the results are presented on a logarithmic scale and that the change in the objective function value for the lower levels of convergence can be small relative to the problem being solved.

The Bayesian Optimisation algorithm has overall good performance and tends to perform better for multimodal functions such as the Rastrigin or Styblinski Tang function. The Bayesian Optimisation featured here, called Dragonfly [37], uses several acquisition functions and initially picks any acquisition function with equal probability. As the optimisation continues, acquisition functions that perform well are picked with a larger probability.

The benchmark results for 10% noise are presented in Fig. 26. Overall the convergence speed of all algorithms is reduced when noise is introduced to the problem. The surrogate model performs better or as good as the other methods, in all nine benchmark functions. The NM algorithm, which showed good performance for some of the noise-free benchmark problems, tends to converge prematurely due to the added noise and even performs worse than the RS algorithm for all tested problems.

The Bayesian method tends to perform well for noisy functions, and it is theorised that this is in part due to the stochastic

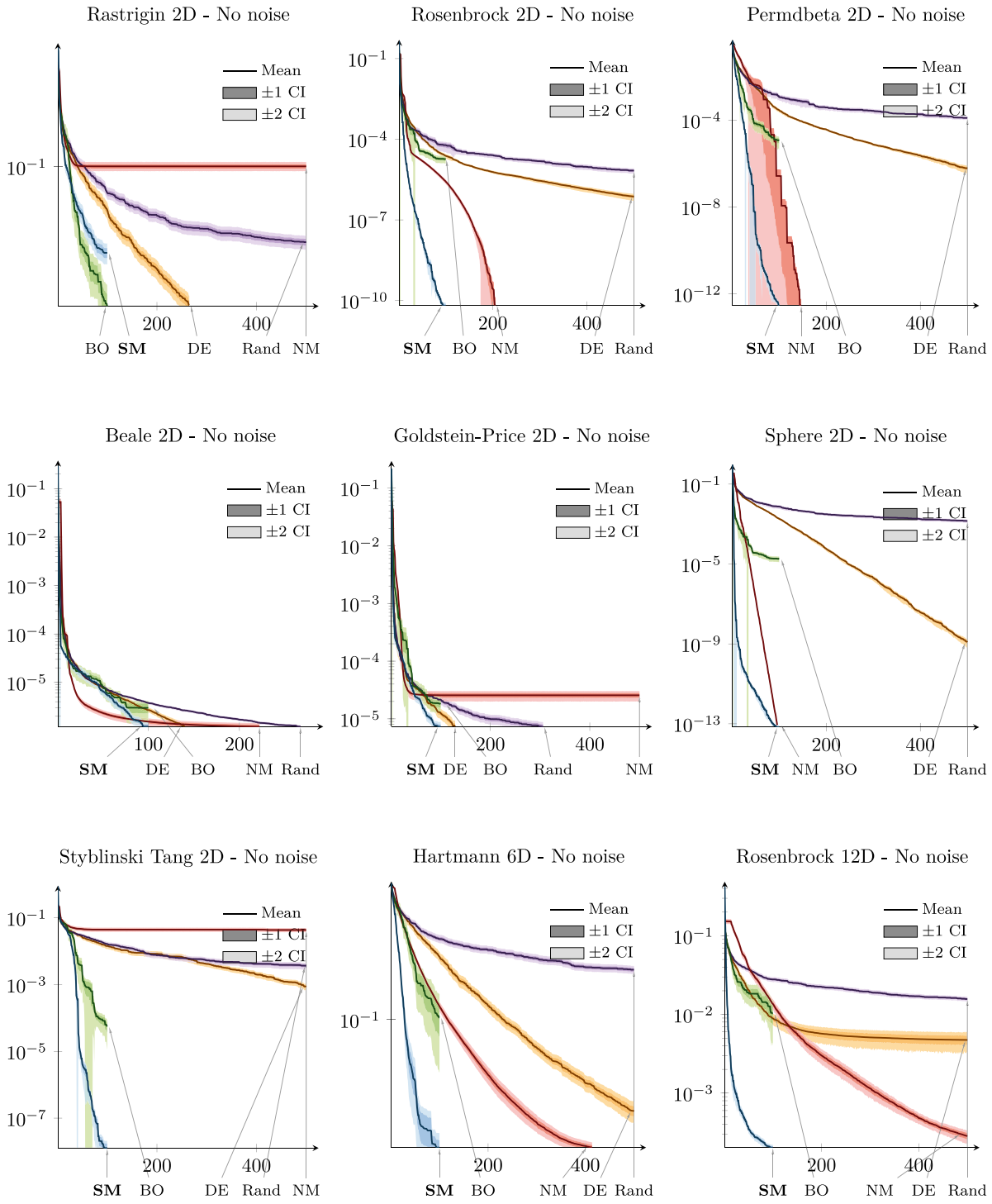


Fig. 25. Optimisation benchmark performance, functions f_{1-9}. Results are normalised by the test function maximum and minimum value.

acquisition function selection. The surrogate model performance might be able to be improved on noisy and multimodal functions by using a similar approach or even a hybrid of the two methods while keeping the current performance for smooth functions.

It should be noted that the Bayesian Optimisation algorithm is the most expensive optimisation algorithm used in the benchmark. The Bayesian algorithm is approximately an order of magnitude more expensive to compute compared to the surrogate model for the investigated test cases. To create the surrogate model, containing 100, 200 and 400 samples, required approximately 3, 12 and 66 s respectively on an Intel(R) Xeon(R) E5-2640 v4 CPU utilising 10 cores. The break-even point between the cost of function calls and the surrogate model creation is quickly outweighed by the reduced number of function calls for expensive function. Considering aerodynamic optimisation, where the function call is a wind tunnel test or a numerical simulation, the wall clock time is large enough to outweigh the surrogate creation cost.

The surrogate model tended to perform particularly well compared to the other methods for functions where the difference between the design dimension is large, such as the Rosenbrock problems.

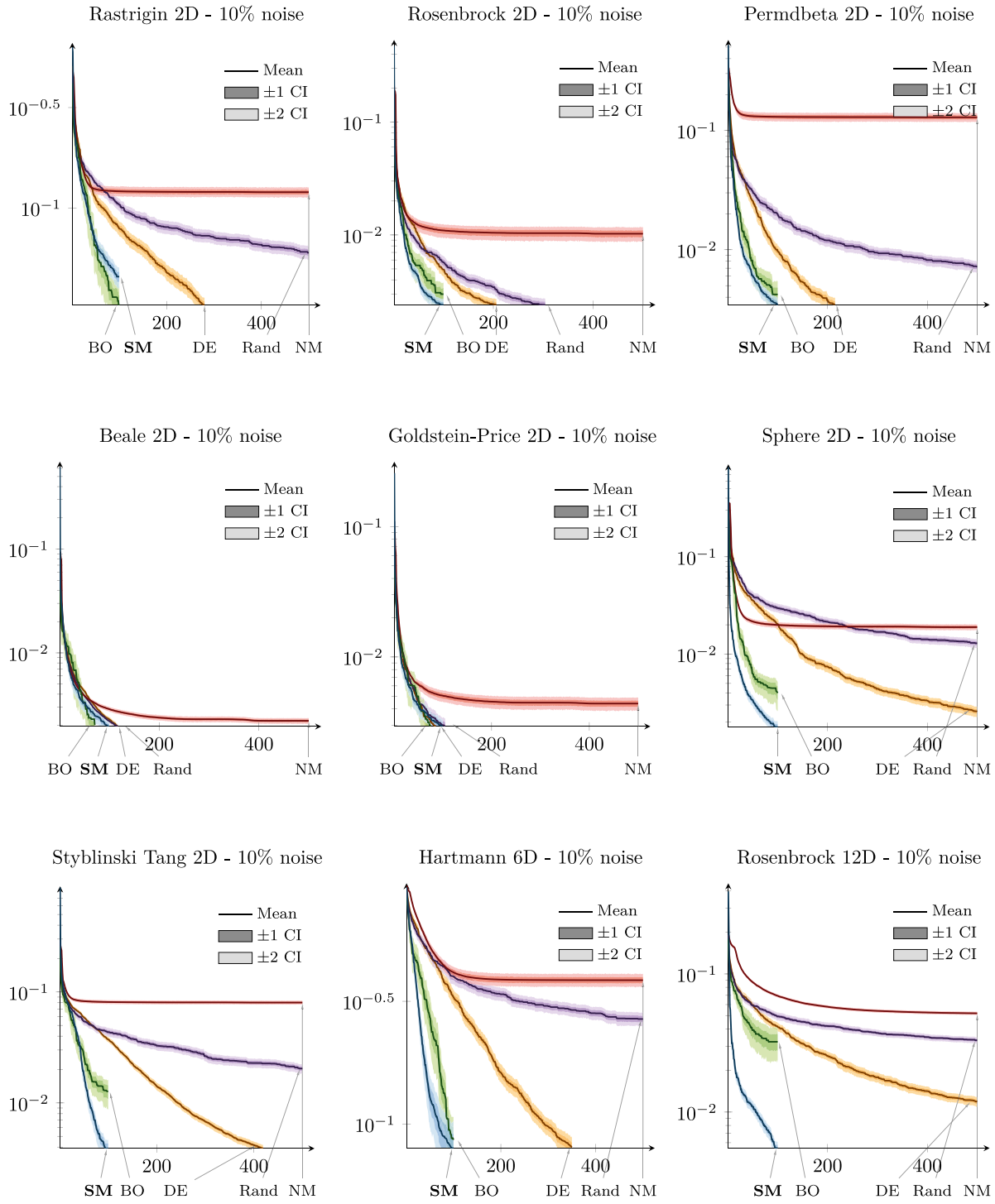


Fig. 26. Optimisation benchmark performance, functions f_{1-9} with 10% white noise. Results are normalised by the test function maximum and minimum value.

Remaining questions of interest for RBF based optimisation is increasing the performance for non-smooth functions such as the Rastrigin function. Kandasamy et al. [37] used an approach where successful objective functions evaluations were selected out of a set of objective functions with a higher probability. Another weakness of the RBF based surrogate model is poor performance around discontinuities in the objective function, which requires further attention. The computational cost of creating the surrogate model is negligible for most aerodynamic optimisation. However, to increase the application range to cheaper objective functions, the computational cost needs to be reduced. Anitescu et al. [40] used a coarse grid to train their neural network initially and stated that the approach increased the robustness as well as significantly reducing the computational effort. It is possible that a similar approach could be adapted to the RBF based technique to reduce the cost of the method without sacrificing accuracy.

The results indicate that the SM optimisation algorithm is robust, high performing, capable of handling a low number of initial samples and can optimise functions with noise.

5. Concluding remarks

The accuracy of a Proper Orthogonal Decomposition-based surrogate model was compared to a force-based surrogate on the generic vehicle DrivAer. The surrogate model performance was compared to Random Sample, Differential Evolution, Nelder–Mead and Bayesian Optimisation on nine test functions with and without added noise. The main outcomes of this study are:

- The force-based surrogate outperformed the POD-based surrogate, although the difference decreased as the number of samples increased.
- The computational cost and storage requirements are larger for the POD-based surrogate compared with the force-based method. The need for generating a vector- or scalar-field can be a decisive factor when deciding between a force- or a POD-based method.
- Adaptive scaling of each design parameter dimension improves the interpolants predictive performance, particularly when the difference in scale between the design inputs and function output is large.
- Adaptive scaling can reduce the surrogate performance penalty associated with increasing the number of design parameters.
- Optimised Latin Hypercube Sampling plans with categorical design parameters improved the overall surrogate model performance, compared to using separate LHC plans for each category.
- The optimisation performance of the surrogate model proved robust when starting from a sparse sampling plan.
- The surrogate-based optimisation algorithm performed better than, or as good as the other algorithms, for 17 out of the 18 investigated benchmark problems.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.asoc.2019.106050.

CRediT authorship contribution statement

Magnus Urquhart: Conceptualization, Investigation, Methodology, Software, Writing - original draft, Writing - review & editing. **Emil Ljungskog:** Conceptualization, Methodology, Software, Writing - review & editing. **Simone Sebben:** Writing - review & editing.

Acknowledgements

The simulations were performed on resources provided by the Swedish National Infrastructure for Computing (SNIC) at High Performance Computing Center North (HPC2N) and National Supercomputing Centre Sweden (NSC) [41].

The authors would like to thank Alexander Broniewicz and Lennert Sterken from Volvo Car Corporation for their support.

Funding

This work is funded by the Swedish Energy Agency grant number P43328-1.

References

[1] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, first ed., Addison-Wesley Longman Publishing Co., Inc, Boston, MA, USA, ISBN: 978-0-201-15767-3, 1989.
[2] S.N. Skinner, H. Zare-Behtash, State-of-the-art in aerodynamic shape optimisation methods, Appl. Soft Comput. (ISSN: 15684946) 62 (2018) 933–962, http://dx.doi.org/10.1016/j.asoc.2017.09.030.
[3] M. Joly, T. Verstraete, G. Paniagua, Differential evolution based soft optimization to attenuate vane–rotor shock interaction in high-pressure turbines, Appl. Soft Comput. (ISSN: 1568-4946) 13 (4) (2013) 1882–1891, http://dx.doi.org/10.1016/j.asoc.2012.12.005.
[4] A. Massaro, E. Benini, A surrogate-assisted evolutionary algorithm based on the genetic diversity objective, Appl. Soft Comput. (ISSN: 1568-4946) 36 (2015) 87–100, http://dx.doi.org/10.1016/j.asoc.2015.06.026.
[5] T. Goel, R. Haftka, W. Shyy, N. Queipo. Ensemble of surrogates, Ensemble of surrogates, Struct. Multidiscip. Optim. 33 (3) (2007) 199–216, http://dx.doi.org/10.1007/s00158-006-0,051-9, ISSN 1615-147X, 1615-1488.
[6] C. Sun, Y. Jin, R. Cheng, J. Ding, J. Zeng, Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems, IEEE Trans. Evol. Comput. 21 (4) (2017) 644–660, http://dx.doi.org/10.1109/TEVC.2017.2675628, ISSN 1089-778X, 1089-778X, 1941-0026.
[7] J. Luo, Design optimization of the last stage of a 4.5-stage compressor using a POD-based hybrid model, Aerosp. Sci. Technol. (ISSN: 12709638) 76 (2018) 303–314, http://dx.doi.org/10.1016/j.ast.2018.01.043.
[8] E. Iuliano, Global optimization of benchmark aerodynamic cases using physics-based surrogate models, Aerosp. Sci. Technol. (ISSN: 12709638) 67 (2017) 273–286, http://dx.doi.org/10.1016/j.ast.2017.04.013.
[9] L. Miretti, E.M. Ribaldone, N. Tobia, N. Paola, Application of POD techniques in CFD optimization of vehicle aerodynamics, in: International Conference on Vehicle Aerodynamics 2016: Aerodynamics By Design, 2016.
[10] V. Dolci, R. Arina, Proper orthogonal decomposition as surrogate model for aerodynamic optimization, Int. J. Aerosp. Eng. 2016 (2016) 1–15, http://dx.doi.org/10.1155/2016/8092824, ISSN 1687-5966, 1687-5974.
[11] E. Iuliano, D. Quagliarella, Proper orthogonal decomposition, surrogate modelling and evolutionary optimization in aerodynamic design, Comput. & Fluids (ISSN: 0045-7930) 84 (2013) 327–350, http://dx.doi.org/10.1016/j.compfluid.2013.06.007.
[12] A. Heft, T. Indinger, N. Adams, Introduction of a new realistic generic car model for aerodynamic investigations, in: SAE 2012 World Congress & Exhibition, SAE International, 2012, http://dx.doi.org/10.4271/2012-01-0168.
[13] J.L. Lumley, The structure of inhomogeneous turbulent flows, in: A.M. Yaglom, V.I. Tararsky (Eds.), Atmospheric Turbulence and Radio Wave Propagation, 1967.
[14] K. Taira, S.L. Brunton, S.T.M. Dawson, C.W. Rowley, T. Colonius, B.J. McKeon, O.T. Schmidt, S. Gordeyev, V. Theofilis, L.S. Ukeiley, Modal analysis of fluid flows: An overview, AIAA J. 55 (12) (2017) 4013–4041, http://dx.doi.org/10.2514/1.J056060.
[15] T. Muld, G. Efraimsson, D. Henningson, Mode decomposition on surface-mounted cube. 88 (3) (2012) 279–310. ISSN 1386-6184, 1573-1987. http://dx.doi.org/10.1007/s10494-011-9355-y.
[16] S. Bates, J. Sienz, V. Toropov, Formulation of the optimal latin hypercube design of experiments using a permutation genetic algorithm. 2011, 2004, http://dx.doi.org/10.2514/6.2004-2011.
[17] S. Jakobsson, B. Andersson, F. Edelvik, Rational radial basis function interpolation with applications to antenna design, J. Comput. Appl. Math. (ISSN: 0377-0427) 233 (4) (2009) 889–904, http://dx.doi.org/10.1016/j.cam.2009.08.058.
[18] K. Price, R. Storn, J. Lampinen, Differential Evolution: A Practical Approach To Global Optimization, in: Natural Computing Series, Springer, Berlin ; New York, ISBN: 978-3-540-20950-8, 2005.
[19] S. Rippa, An algorithm for selecting a good value for the parameter c in radial basis function interpolation, Adv. Comput. Math. 11 (2–3) (1999) 193–210, http://dx.doi.org/10.1023/A:1018975909870, ISSN 1019-7168, 1572-9044.
[20] K.C. Giannakoglou, Design of optimal aerodynamic shapes using stochastic optimization methods and computational intelligence, Prog. Aerosp. Sci. (ISSN: 0376-0421) 38 (1) (2002) 43–76, http://dx.doi.org/10.1016/S0376-0421(01)00019-7.
[21] P. Chandrashekarappa, R. Duvigneau, Radial Basis Functions and Kriging Metamodels for Aerodynamic Optimization, Research Report RR-6151, INRIA, 2007.
[22] G. Fasshauer, Meshfree Approximation Methods with MATLAB, World Scientific Publishing Co., Inc., River Edge, NJ, USA, ISBN: 978-981-270-634-8, 2007.
[23] K. Burnham, D. Anderson, Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach, second ed., Springer, New York, ISBN: 978-0-387-95364-9, 2002.

[24] J. Mayer, K. Khairy, J. Howard, Drawing an elephant with four complex parameters, Amer. J. Phys. 78 (6) (2010) 648–649, http://dx.doi.org/10.1119/1.3254017.
[25] O. Abedinia, N. Amjady, Short-term load forecast of electrical power system by radial basis function neural network and new stochastic search algorithm, Int. Trans. Electr. Energy Syst. (ISSN: 2050-7038) 26 (7) (2016) 1511–1525, http://dx.doi.org/10.1002/etep.2160.
[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (2014) 1929–1958.
[27] J. Bezanson, A. Edelman, S. Karpinski, V. Shah, Julia: A fresh approach to numerical computing, SIAM Rev. (ISSN: 0036-1445) 59 (1) (2017) 65–98, http://dx.doi.org/10.1137/141000671.
[28] M. Urquhart, Latinhypercubesampling.jl - release v1.2.0, 2018, https://github.com/MrUrq/LatinHypercubeSampling.jl.
[29] M. Urquhart, Properorthogonaldecomposition.jl - release v0.1.0, 2018, https://github.com/MrUrq/ProperOrthogonalDecomposition.jl.
[30] E. Ljungskog, Scatteredinterpolation.jl - release v0.3.0, 2018, https://githubcom/eljungsk/ScatteredInterpolation.jl.
[31] R. Feldt, Blackboxoptim.jl - release v0.4.0, 2018, https://github.com/robertfeldt/BlackBoxOptim.jl.
[32] M. Urquhar, Surrogatemodeloptim.jl - release v0.4.1, 2019, URL https://doi.org/105281/zenodo.3549619.
[33] E. Ljungskog, S. Sebben, A. Broniewicz, C. Landström, On the effects of wind tunnel floor tangential blowing on the aerodynamic forces of passenger vehicles, SAE Int. J. Passeng. Cars - Mech. Syst. (ISSN: 1946-4002) 10 (2) (2017) http://dx.doi.org/10.4271/2017-01-1518.
[34] P. Mogensen, A. Riseth, Optim: A mathematical optimization package for Julia - optim.jl release v0.17.2, J. Open Source Softw. 3 (24) (2018) 615, http://dx.doi.org/10.21105/joss.00615.
[35] J.A. Nelder, R. Mead, A simplex method for function minimization, Comput. J. 7 (4) (1965) 308–313, http://dx.doi.org/10.1093/comjnl/7.4.308, ISSN 0010-4620, 1460-2067.
[36] F. Gao, L. Han, Implementing the nelder-mead simplex algorithm with adaptive parameters, Comput. Optim. Appl. 51 (1) (2012) 259–277, http://dx.doi.org/10.1007/s10589-010-9329-3, ISSN 0926-6003, 1573-2894.
[37] K. Kandasamy, K.R. Vysyaraju, W. Neiswanger, B. Paria, C. Collins, J. Schneider, B. Poczos, E. Xing, Tuning hyperparameters without grad students: Scalable and robust Bayesian optimisation with dragonfly, 2019, arXiv:1903.06694 [cs, stat], arXiv:1903.06694.
[38] P. Frazier, A tutorial on Bayesian optimization, 2018, arXiv:1807.02811 [cs, math, stat], arXiv:1807.02811.
[39] S. Sarra, Y. Bai, A rational radial basis function method for accurately resolving discontinuities and steep gradients, Appl. Numer. Math. (ISSN: 01689274) 130 (2018) 131–142, http://dx.doi.org/10.1016/j.apnum.2018.04.001.
[40] C. Anitescu, E. Atroshchenko, N. Alajlan, T. Rabczuk, Artificial neural network methods for the solution of second order boundary value problems, Comput. Mater. Continua (ISSN: 1546-2226) 59 (1) (2019) 345–359, http://dx.doi.org/10.32604/cmc.2019.06641.
[41] SNIC, Swedish national infrastrucutre for computing, 2018.